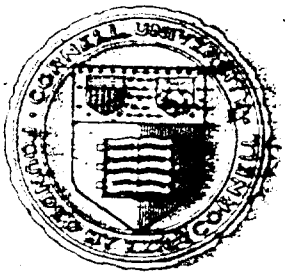

A Plane-Sweep Algorithm for Exact Simulation of a Quasi-Static Mechanical System of Compliantly-Connected Rigid Bodies (Extended Abstract)

Bruce R. Donald*
Dinesh K. Pai**

TR 90-1177
December 1990

TECHNICAL REPORT

Department of Computer Science
Cornell University
Ithaca, New York

2

A Plane-Sweep Algorithm for Exact Simulation of a Quasi-Static Mechanical System of Compliantly-Connected Rigid Bodies (Extended Abstract)

Bruce R. Donald*
Dinesh K. Pai**

TR 90-1177
December 1990

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Supported in part by the National Science Foundation under Grants No. IRI-8802390, IRI-9000532, and by a Presidential Young Investigator Award No. IRI-8957316, and in part by the Mathematical Sciences Institute.
**Supported in part by ONR Grant N00014-88K-0591, ONR Grant No. N00014-89J-1946, and NSF Grant DMC-86-17355.

A Plane-Sweep Algorithm for Exact Simulation of a Quasi-Static Planar Mechanical System of Compliantly-Connected Rigid Bodies (Extended Abstract)

Bruce Randall Donald* **Dinesh K. Pai†**
Computer Science Department, Cornell University

Abstract

Automatic fastening (eg, snap fasteners) is a central theme in CAD, manufacturing and design for assembly. Snap fasteners are non-rigid bodies, and hence predicting the motion of flexible objects during an assembly is of interest as an algorithmic problem. We consider a planar system of rigid bodies that are compliantly connected by torsional springs. A single "root" body undergoes pure translation, during which k passively rotationally compliant members called "pawls" can contact with immovable planar obstacles. As the motion proceeds, the pawls deform (deflect) around the obstacles in response to the kinematic constraints and the frictional contact forces. The pawls can also "snap" off an obstacle edge towards their zero position; this motion is modeled using a pure rotation. The *simulation problem* is to determine the motion of the pawls as parameterized by the translation distance of the root, and to compute the termination configuration of the system if it is reachable. The solution trajectory Φ of the system is piecewise cubic in the configuration space, and the dynamics may be reduced to erecting cubic "local" configuration space constraints. The simulation never leaves a connected component F of configuration space. We can compute the solution Φ by sweeping k planar slices of F (each slice is a planar arrangement of cubic curves). Each slice has complexity $O(\lambda_r(n))$, and can be constructed efficiently using a red-blue merge algorithm (r is a small constant). Reducing motion prediction to sweeping a planar arrangement of curves allows us to solve the simulation problem exactly in $O(k\lambda_r(n)\log^2 n)$ time.

*Supported in part by the National Science Foundation under grants No. IRI-8802390, IRI-9000532 and by a Presidential Young Investigator award No. IRI-8957316, and in part by the Mathematical Science Institute.

†Supported in part by ONR Grant N00014-88K-0591, ONR Grant N00014-89J-1946, and NSF Grant DMC-86-17355.

1 Introduction

Previous work on algorithmic motion planning has largely concentrated on the movement problem for rigid bodies [LoP,Don,Yap]. Recently, work in compliant motion planning under uncertainty [LMT, Erd, CR, Don2, Don3, Can2, Bri, FHS] has focused on the problem of moving rigid objects (e.g. pegs) in contact (i.e., compliantly) with obstacles (such as holes) under force-control, subject to bounded uncertainty and error. The only combinatorially precise results that have been obtained for compliant motion have been for pure translations. Hence they are essentially inapplicable for any real systems, which typically can rotate. Indeed, planning and simulation for systems with rotational bodies and rotational compliance has resisted solution in the sense that only approximate, heuristic, or numerical algorithms are known. We consider a mechanical system of bodies that can compliantly slide on obstacles while they translate and rotate in the plane. We provide an exact, efficient algorithm for predicting the motion of these rather interesting devices, which, as we we discuss below, are important in manufacturing, design, and factory assembly. The key ideas we use are: red-blue merge algorithms, a simple dynamical systems model, and local dynamic constraints. These tools permit us to reduce the simulation to a plane sweep of a " dynamically annotated" slice of configuration space. We hope that the paradigm of "simulation as sweep" may be useful in other domains.

We are pursuing an algorithmic theory of *design for assembly.* To this end we are developing and implementing algorithms that can analyze and generate designs for objects so that they will be easy to assemble. In particular, we observe that real objects that robots might assemble are typically not rigid. For example, a Sony Walkman is made of plastic parts that snap together. Significant advances were made in the design of the IBM ProPrinter, by replacing traditional fasteners such as screws with plastic parts that simply snap together. The reason these plastic parts snap together is that they are *flexible*: more precisely, they are *passively compliant.* This means that when the parts are brought together and an external force applied, the parts deform in a prescribed way. More interestingly, the force required to mate two parts may be much less than the force required to take them apart.

Since we wish to be able to design and have our robots assemble such objects given task-level descriptions, we must have a systematic program for reasoning about and predicting their motions in contact. To this end, Pai and Donald [PD, DP], made precise a sufficiently powerful notion of flexibility to model the objects above, which encompasses several important and complicated mechanisms in mechanical design and automated assembly: snap-fasteners, latches, ratchet and pawl mechanisms, and escapements (see fig. 1). We modeled the physics of interaction between the flexible parts and the environment (including their mating parts). Using these tools, we can proceed to develop combinatorially precise geometric algorithms for predicting the motion of a flexible object near and in contact with its mating part.

We view the simulation problem, even with rotational compliance and quasi-static mechanics, as a problem that can be solved by careful reduction to a plane sweep. In particular, for each "pawl." we reduce to sweeping an planar arrangement of algebraic curves of low degree. The connected component of free space defined in this planar arrangement has complexity $O(\lambda_r(n))$, and can be constructed in time $O(\lambda_r(n)\log^2 n)$ using a red-blue merge algorithm [GSS]. Here $\lambda_r(n)$ is the (almost linear) maximum length of (n, r) Davenport Schinzel sequences [GSS]. r is a small constant related to the number of times two cubic[1] configuration space constraint curves can intersect. Our

[1]By "cubic" we mean the total degree of the defining multinomial is 3. Our curves have additional structure, such

approach [PD, DP] to modeling rotational compliance and to incorporating frictional constraints leads to the first formulation of the simulation prediction problem which permits a *reduction* of motion prediction to plane sweep. Our solution differs from previous work on predicting, bounding, and planning rotationally compliant motions with quasi-static mechanics in that it is (i) *purely algebraic*, and hence exact, (ii) *combinatorially precise*, in that the computational complexity is exactly known, and (iii) requires no integration.[2]

1.1 Problem Statement

We consider the problem of moving a flexible, linked body $\mathcal{M}$ in the plane, in a polygonal environment $\mathcal{N}$. The flexible body $\mathcal{M}$ is constructed as follows: polygons $\mathcal{M}_h$, $h = 1, \ldots, k$, called "pawls", are attached to a root polygon $\mathcal{M}_0$, at hinge points P_h. Each hinge is coupled with a spring of stiffness κ_h. (See Figure 2).

The motion of the body $\mathcal{M}$ consists of rigid translation of the root polygon $\mathcal{M}_0$. The pawls $\mathcal{M}_h$ are free to move compliantly as dictated by interactions with the environment and the spring.

Time T will be represented by the nonnegative real numbers. A *solution trajectory* Φ for the system $(\mathcal{M},\mathcal{N})$ is a family of maps $(\phi_0, \phi_1, \ldots, \phi_k)$ where $\phi_0 : T \rightarrow \Re^2$ specifies the configuration of the root and each $\phi_h : T \rightarrow S^1$ specifies the orientation of pawl polygon $\mathcal{M}_h$, $h = 1, \ldots, k$. Hence the global configuration of pawl $\mathcal{M}_h$ at time t is given by $(p(t), \phi_h(t))$, where $p(t)$ differs by a constant offset from $\phi_0(t)$ (see eq. (1) below). We see that the configuration space of the system $(\mathcal{M},\mathcal{N})$ is[3] $C = \Re^2 \times \overbrace{S^1 \times \cdots \times S^1}^{k\ S^1\text{'s}}$.

The *Simulation Problem* is to determine:

1. The solution trajectory $\Phi : T \rightarrow C$ of the system.
2. The time of termination of the motion, the cause of termination (such as sticking due to friction, sticking due to kinematic constraints, etc.), and the configuration of $\mathcal{M}$ at termination.
3. The time history of contacts between $\mathcal{M}$ and $\mathcal{N}$.

Some extensions of the simulation problem are considered in [DP], including the determination of the time history of forces during the motion, and the effect of uncertainty.

We make the following assumptions about the physics of object interactions and the motion of $\mathcal{M}$:

- Object interactions are restricted to those between $\mathcal{M}_h$ and $\mathcal{N}$. In other words, pawls do not collide with each other, but may collide with the environment $\mathcal{N}$. The effect of this assumption is to make the motion of each pawl independent of the motion of other pawls. Henceforth we shall consider the motion of a single pawl $\mathcal{M}_h$.

---

as low-degree parameterizations, as well.

[2]Note that [Don2,Can2,Bri,FHS] address geometric reachability issues for translationally compliant objects, but these objects cannot rotate.

[3]This is the configuration space with no springs, but only kinematic constraints. Introduction of springs means that it is not possible to identify a rotation of 2π with 0, since at 2π the pawl is "cocked". Hence the introduction of dynamics forces us to pass to the covering space $\Re^{k+2}$. Our analysis goes through *mutatis mutandis* for the covering space.

- Since the root $\mathcal{M}_0$ is undergoing a rigid translation, so does the hinge point P_h. We shall assume that this translation is a straight line motion given by

$$p(t) = p_o + \dot{p}t \tag{1}$$

where t is the time, p_o is the initial position (at $t = 0$) of P_h, and $\dot{p}$ is its velocity.

- Stable contact: Suppose the pawl is in contact with a feature (edge or vertex) of the environment (for example, during sliding). We assume that if we perform a small displacement of the pawl away from the environment, the torque on the pawl due to the spring is such that the contact will be restored. This assumption is no. very restrictive at all – in fact. in the face of even the smallest uncertainty, stable contacts are the only ones one can hope to observe in practice.

- Quasi-static motion: The motion is assumed to be slow enough that inertial effects are not significant. This corresponds to assuming that there is no acceleration of the pawl, and hence the forces on the pawl are balanced. The quasi-static assumption is reasonable at small speeds and is widely used (see, for example, [Whi,Ma,Erd,Don2]).

- If the pawl slides off $\mathcal{N}$ into free space, it may have a residual torque due to the spring being cocked. We shall assume that the pawl rotates back towards its rest orientation at such great speed that p does not change significantly during the rotation and can be taken to be constant. This, incidentally, is the "snap" in the snap-fasteners that we wish to model. This assumption may appear to contradict the assumption of quasi-static motion, but is in fact practically a consequence. Quasi-static motion implies that the root is moving "slowly-enough" for the forces to be balanced. Hence, when a pawl is in free space and has a residual torque, its motion can be fast compared to that of the root, resulting in a "snap". This assumption can be relaxed by assuming a linear relationship between the translation p and the rotation (e.g.. see [Can]), but we do not deal with it here.

- The forces of friction arising from contact obey Coulomb's Law. We further assume that there is a single coefficient of friction. This assumption is also widely used.

These assumptions define a simple but adequate dynamical system. We will exploit the geometry of this system extensively to obtain our results.

Two types of contact are possible between the pawl $\mathcal{M}_h$ and a polygon in $\mathcal{N}$. Following the convention of Lozano-Pérez [LoP] and Donald [Don], we say that Type-A contact occurs when a vertex of $\mathcal{N}$ touches an edge of the pawl; Type-B contact occurs when a vertex of the pawl touches an edge of $\mathcal{N}$.

We can now write the contact constraint equations for the two types of contact, as in [Can]. We shall index features (vertices and edges) of the moving pawl by the subscript i and features of the polygonal environment by the subscript j. Let an edge of $\mathcal{M}_h$ be represented by its outward normal, n_i, and its distance to the hinge point along the normal, d_i. Let p_j be a vector to the contact vertex of $\mathcal{N}$. Let R_θ be the linear transformation which rotates a vector by an angle θ. Then the type-A constraint can be written as

$$(p_j - p)\cdot R_\theta n_i - d_i = 0. \tag{2}$$

Similarly, the type-B constraint can be written as

$$(R_\theta p_i + p)\cdot n_j - d_j = 0. \tag{3}$$

Where p_i is the vector from the hinge point P_h to the contact vertex of $\mathcal{M}_h$. The derivation of eqs. (2) and (3) has been discussed extensively in the literature, eg., [LoP, Don, Can].

2 Statement of Results

2.1 Exact solutions for mechanical simulations

The approach in [DP] deviated significantly from earlier simulation methods. A major impediment to developing simulation systems has been the apparent necessity to integrate out the differential mechanics in order to determine the long-term behavior of the system. This problem is exacerbated by the fact that in many models of rotational compliance such as the generalized damper (eg., [LMT, Erd, Don2, Don3, Can2]), the resulting trajectories are not known to be algebraic; neither do we have ways of computing algebraic bounding approximations (or forward projections). Hence the traditional numerical approach to simulating such systems has been the following:

Typical Simulation Algorithm

1. Given a state x of the system, numerically integrate the differential equation governing motion of the system. Step forward in time to obtain (approximately) new state x'.
2. Perform collision detection either at x' or along the path from x to x'.
3. If the constraints have changed, reformulate the differential equation.
4. Repeat.

Numerical simulation of mechanical systems is fraught with error, special cases, and numerical problems. They are rarely combinatorially precise, and almost never come with guarantees of accuracy (see [CDRX] for exceptions). We show how in the case of our system, numerical simulation can be avoided, and exact solutions can be obtained. We cannot claim that this can be done in general. However, our method yields, in this case, computationally efficient, exact solutions, and may possibly be useful in other domains.

2.2 Computational complexity

The focus of our initial work [DP] was on modeling and robotics issues, and not on the computational geometric aspects. Our first algorithm, while polynomial, was naïve. In this paper we give a considerably faster algorithm by mounting a computational geometric attack. In addition, the new algorithm sheds light on several interesting combinatorial and structural issues. Furthermore, it leads to a systematic classification of special cases, and has connections to the qualitative analysis of dynamical systems.

More specifically: In [DP], we showed how our model of compliance permits us to obtain algebraic, closed-form solutions to simulation problems for a rotationally compliant object, and how this leads to *exact* algorithms for analyzing designs for assembly. In particular, the linear map $p(t)$ is given by eq. (1), and once we "rationalize" rotations via the standard substitution $u = \tan\frac{\theta}{2}$, each map ϕ_h is piecewise-cubic. The naïve algorithm can be summarized as follows.

Naïve Exact Algorithm [DP]

1. Given a state x of the system, possibly subject to a configuration space constraint f (f has form eq. (2) or (3)), calculate a local solution trajectory Φ respecting f. Φ will be a linear or cubic curve in the configuration space.

2. The configuration space constraints are algebraic surfaces of bounded degree. Intersect Φ with the surfaces to perform exact collision detection.

3. A collision results either in termination of the motion or in a change of constraint. Update f if necessary.

4. Repeat.

Suppose the obstacles have m vertices and the moving object (root and one pawl) has l vertices. Let $n = ml$ be the measure of the geometric complexity. In [DP] we were able to show that this naïve algorithm runs in time $O(n^2 \log n)$. For k pawls, the complexity of the naive algorithm was $O(kn^2 \log n)$.

In this paper, we give a new simulation algorithm that is also exact, and runs in time $O(k\lambda_r(n)\log^2$ Our method reduces the simulation problem to a plane sweep of an arrangement of algebraic curves in configuration space. To obtain this result we prove the following points. For simplicity, assume below that $k = 1$, and hence configuration space $C = \Re^2 \times S^1$.

1. The simulation never leaves a connected component F_0 of free configuration space.

2. The solution trajectory Φ of the system is piecewise cubic in the configuration space, and the dynamics may be reduced to erecting cubic "local" configuration space constraints.

3. Translational motion of the root polygon (eq. (1)) restricts the reachable configurations to a 2D cylinder $Y \subset C$.

4. Y embeds in C as follows:

$$\begin{array}{ccc} Y & \hookrightarrow & \Re^2 \times S^1 \\ (t,\theta) & \mapsto & (p(t),\theta). \end{array} \tag{4}$$

   where t moves along the axis of the cylinder. Hence we view Y as $\Re \times S^1$. The time evolution $t \in T$ of the system corresponds to the $\Re$ factor. Hence the solution "sweeps" along the cylinder Y in the direction of the axis.

5. The configuration space constraints are manifest as cubic curves on Y.

6. As we sweep the cylinder Y, a simple dynamical system models the motion of the configuration point on the configuration space constraints. The orbits of this system are piecewise algebraic of bounded degree.

7. Parameterize Y to the plane (this only takes 2 charts). Then $F = Y \cap F_0$ is defined by a planar arrangement of cubic curves.

8. F has size $O(\lambda_r(n))$ and can be constructed in time $O(\lambda_r(n)\log^2 n)$ using a red-blue merge algorithm, as described in [GSS].

9. We can compute the solution Φ by a plane-sweep of the slice F, i.e., by sweeping a planar arrangement of cubic curves. This allows us to solve the motion prediction problem exactly in $O(\lambda_r(n)\log^2 n)$ overall time.

Hence we prove

Theorem 2.1 *The* Simulation Problem *defined above can be solved in time $O(\lambda_r(n)\log^2 n)$ and space $O(\lambda_r(n))$, where r is a small constant.*

3 Details, Analysis, and Description of our Approach

3.1 Computing the Connected Component of Free Space

We can only sketch the idea of our proofs here. Consider the motion of a single pawl $\mathcal{M}_h$, whose translation is governed by eq. (1). As is well known, if we "rationalize" rotations using the standard substitution $u = \tan\frac{\theta}{2}$, the n constraints given by eqs. (2) and (3) are manifest as algebraic ruled surfaces $\{f_1, \ldots, f_n\}$ in a 3D configuration space with coordinates (x, y, u). These constraints are simultaneously linear in the position parameters x and y and quadratic in the rotational parameter u. Each surface is only "applicable" for some range of orientations $[u_0, u_1]$, by which we effectively mean that the surface only "exists" for u in this range (see [Don]). See fig. 3, from [Bro].[4]

Let $\hat{u}$ be a vector along the u-axis (think of $\hat{u}$ as $(0, 0, 1)$). Now, the constraint of pure translation (eq. (1)) of the hinge point P_h of the pawl $\mathcal{M}_h$ restricts any possible evolution of the system to lie in a "plane" (2D subspace) of (x, y, u)-space. We call this "plane" P_Y; it has "normal" $\hat{u} \times (\dot{p}, 0)$, and it corresponds to a chart for the cylinder Y discussed in sec. 2.2. We note that furthermore, at time t, the state of the system is constrained to lie on a line $L(t)$ parallel to $\hat{u}$ in the plane P_Y. This is the line of points $(p(t), u)$, for $u \in \Re$. The line $L(t)$ sweeps across the plane in direction $(\dot{p}, 0)$ as t increases, and thus we call $L(t)$ the *sweep line.*

Hence, a natural coordinate system for the plane P_Y is given by (t, u). As time t increases, the vertical line $L(t)$ sweeps across P_Y. This line contains the state of the system. It is our task to calculate the u coordinate as t evolves (i.e., increases). Now, P_Y has degree 1 and hence when intersected with a constraint f_i we obtain a cubic[5] curve segment γ_i in the (t, u) plane. So all the configuration space constraints are manifest as an arrangement of cubic curve segments $\{\gamma_1, \ldots, \gamma_n\}$ in this plane, where $\gamma_i = f_i \cap P_Y$ $(i = 1, \ldots, n)$. In our algorithm, the sweep line sweeps across this planar arrangement of curves, and as we sweep, we compute the trajectory of the system. "Events" caused by crossing the curves γ_i will modify the trajectory, as we discuss below.

A priori, the arrangement of curves can have complexity $O(n^2)$, and in fact, the set of free configurations in P_Y can also have size $\Omega(n^2)$ in the worst case (see [KS,GSS]). However, we can directly apply the results of [GSS] as follows. We note that the system begins at some configuration $z_0 \in P_Y$. z_0 lies in one connected component $F \subset P_Y$ of free space and the resulting path can never leave F since it corresponds to a physical simulation. [GSS] show that:

[4] We thank R. Brost for providing us with these figures, from [Bro].

[5] In fact, each curve γ_i is simultaneously quadratic in u and linear in t.

Lemma 3.1 [GSS] *The combinatorial complexity of F is $O(\lambda_{s+2}(n))$, and we may precompute it (i.e., compute it before our plane sweep) in time $O(\lambda_{s+2}(n)\log^2 n)$. Here s is a small constant bounding the number of times that two configuration space constraint curves γ_i and γ_j can intersect.*

Recall $\lambda_r(n)$ is the (almost linear) maximum length of (n, r) Davenport Schinzel sequences [GSS], [HS,ASS]. It is very likely that for a wide variety of situations encountered in practice (see, eg., [Bro]) that $s \leq 1$. However, it is certain that a worst case bound is $s \leq d^2$ for curves of degree d. Note that $s \leq 1$ would tighten our bound to space $O(n\alpha(n))$ and time $O(n\alpha(n)\log^2 n)$ [HS].

3.2 Sweeping the Connected Component of Free Space

3.2.1 Kinematics

We postpone our discussion of friction until sec. 3.3. Here we treat the frictionless case first, defining six types of local geometric events, called "sweep events" that will be detected and handled during the plane sweep. These events are purely kinematic, i.e., they do not depend on friction.

Having computed the connected component $F \subset P_Y$ containing the initial configuration, we now sweep F with the line $L(t)$ in the (t, u)-plane. By an abuse of notation we will now let γ_i, γ_j etc. denote the curves bounding F that we precomputed (along with their intersections) in sec 3.1. We now define the following sweep events: (i) *translational collision,* (ii) *sliding collision,* (iii) *jamming due to incompatible kinematics.* The sweep algorithm will detect sweep events. Each event will be *handled*, by which we mean that the solution trajectory we compute may be modified. In between events, the trajectory is piecewise algebraic.

First suppose there are no obstacles. Then the trajectory of the system will stay at $u = 0$ in the (t, u)-plane (i.e., the orientation of the pawl will not change). Call the point $z(t)$ on $L(t)$ representing the state of the system the *sweep point.* So $z(t)$ is the solution trajectory we compute.

Now, in the presence of obstacles, sweep events occur as the the sweep line crosses the curves $\{\gamma_i\}$. These events are enumerated in figs. 4-6. We can explain the trajectory computation algorithm like this: the dynamical system described in sec. 1.1 has the following geometric interpretation in slice P_Y. As the sweep line $L(t)$ crosses the (t, u), plane, the the u-coordinate $u(t)$ of the sweep point $z(t) \in L(t)$ moves. In the plane P_Y, the line $u = 0$ is an *attractor*, and we imagine a vector field on P_Y parallel to the u-axis and pointing towards the t-axis. Hence the attracting vectors are parallel to $-\hat{u}$ for $u > 0$ and parallel to $+\hat{u}$ for $u < 0$. The curves γ_i act as (holonomic) constraints. The sweep point cannot cross these curves, but it can follow them as $L(t)$ moves. They can prevent motion of the sweep point from attaining $u = 0$.

For example, see fig. 4. If the trajectory is at the $u = 0$ position and the sweep point $z(t)$ encounters a constraint γ_i, then the sweep point complies to the constraint and is forced to move away from the zero line (u becomes positive here). This corresponds to a pure translational collision, followed by a continued motion of the root which "cocks" the pawl against an obstacle. During this motion, the sweep point follows γ_i. If a new constraint γ_j is reached, then the sweep point slides along the curve γ_j in turn. This corresponds to a sliding collision: while sliding on constraint γ_i, the pawl hits constraint γ_j. The motion continues, following γ_j compliantly. Hence the sliding collision can result in a constraint change. Finally, if the sweep point is following a curve γ_i which crosses $u = 0$, the trajectory breaks contact there and continues along the t-axis. This event is a "dual" subcase of type (i).

As can be seen from fig. 4, some constraint changes result in jamming due to incompatible kinematics. This occurs as follows. Define the outward normal η_i of a curve γ_i to point into free space F. Let $\hat{t}$ be a unit vector in the positive t-direction. Jamming occurs at $\gamma_i \cap \gamma_j$ when both the inner products

$$\eta_i \cdot \hat{t} \text{ and } \eta_j \cdot \hat{t} \tag{5}$$

are negative. At this point the simulation is terminated, because further motion is impossible.

Pure translational collision events can occur where a curve γ_i intersects the line $u = 0$. *Sliding collisions* can occur when two boundary curves of F intersect, i.e., at $\gamma_i \cap \gamma_j$. Jamming events can occur when both normals at $\gamma_i \cap \gamma_j$ point in the $(-t)$-direction. A non-jamming sliding collision causes a change of constraint (i.e., the sweep point now follows γ_j instead of γ_i). It is clear that sweep events of type (i), (ii), and (iii) are local geometric conditions and can be detected and handled while sweeping the line $L(t)$ over F. Similarly, it is clear that modifying the trajectory $z(t)$ at a sweep event can be done in $O(1)$ time.

3.2.2 Snapping Free or Jamming on A Single Constraint

We now define the sweep events (iv) *snapping free from* and (v) *jamming on a single constraint*. Suppose the sweep point is following a constraint curve γ. A singularity occurs at vertical tangencies of γ. See fig. 5. Assume wlog that γ lies in the halfplane $u > 0$. There are two possibilities. If the F is concave at the singularity, then the sweep point has been following the "upper" branch of the curve. After the singularity, the sweep point follows the vector field attracting it towards $u = 0$. That is, the sweep point moves parallel to the u-axis toward the t-axis. It stops at the first new constraint curve it hits while moving away from the singularity towards the line $u = 0$. If no constraints are encountered, it stops at $u = 0$. This motion corresponds to the pawl "snapping free" from a single constraint edge. It executes an instantaneous pure rotation towards the zero position. If another constraint is in the way, then it stops there.

If F is convex at the singularity, then no further motion is possible, and the motion jams there on a single constraint. At this point the simulation is terminated.

Clearly, singularity (vertical tangency) is a local geometric condition that can be detected during the plane sweep of F, since each curve is algebraic.

There is one more kinematic sweep event that is "dual" to type (iii) jamming due to incompatible kinematics. It is type (vi) *snapping free from a vertex*. It occurs at a constraint change $\gamma_i \cap \gamma_j$ (i.e., the sweep point is following a curve γ_i, and it hits another curve γ_j). However, in this case, both the outward normals η_i and η_j point in the positive t-direction. That is, the dot products in eq. (5) are both positive. In this case, the sweep point "snaps free" from $\gamma_i \cap \gamma_j$ and moves vertically towards the attractor $u = 0$. The snapping free happens just as in event (iv) above. Snapping free from a vertex corresponds to the situation where suddenly there are no holonomic constraints on the pawl, so it can move towards its rest position $u = 0$. Conceptually, there is little difference from event (iv) (snapping free from one constraint).

It is clear that sweep events of type (iv), (v), and (vi) are local geometric conditions and can be detected and handled while sweeping the line $L(t)$ over F. It is clear that modifying the trajectory $z(t)$ at a sweep event can be done in $O(1)$ time. To see that six event types suffice, simply enumerate the ways γ_i can (a) intersect γ_j, (b) intersect $u = 0$, or (c) become vertical. Hence we have,

Lemma 3.2 *There are six types of kinematic sweep events, as described above. There are $O(\lambda_{s+2}(n))$ such events overall. Each event is a local geometric condition that can be detected and handled in $O(1)$ time. The output trajectory is piecewise algebraic with at most $O(\lambda_{s+2}(n))$ pieces and degree at most 3.*

Corollary 3.3 *A plane sweep of F that handles all kinematic sweep events can be performed in time $O(\lambda_{s+2}(n)\log\lambda_{s+2}(n))$. This sweep solves the frictionless simulation problem (given F) for a single pawl.*

3.3 Friction

We now briefly describe how friction is handled. From the analysis in [DP], the following is clear: for each configuration space surface f_i we can define two constraints g_i and h_i which are also algebraic surfaces of the same degree as f_i. g_i and h_i depend on the direction of assembly $\dot{p}$ in (1).

The surfaces g_i and h_i break up f_i into *sliding* and *sticking* regions. We call these *qualitative dynamical regions (QDR's)*, by analogy with [BD]. In a sliding region, motion is possible as t increases. In a sticking region, equilibrium results, and no further motion is possible (compare work on translational compliant motion, eg, [Don2, Bri]). The analysis to obtain this result is based on an interesting mechanical analysis which we cannot include for reasons of space; however, see [DP]. Now, when we intersect f_i with the plane P_Y to obtain a curve γ_i (see fig. 6) we obtain a 1D slice of these qualitative dynamic regions (sliding and sticking). Now, the Bezout bound gives an a priori $O(1)$ bound on the number of QDR's per surface. However, in fact, the special structure of our constraints ensures that there will be at most 3 QDR's per connected curve γ_i on the boundary of F. Type-B constraints have (at most) one sliding region surrounded by two sticking regions. Type-A constraints have (at most) one sticking region surrounded by two sliding regions.

Now, we define a seventh type of sweep event, (vii) a *sticking event* as follows. Suppose the sweep point is following a curve γ_i. If it enters a sticking region on the curve, then equilibrium is reached and the simulation is terminated. Entry into the sticking region corresponds to crossing another algebraic curve h_i or g_i, and hence is a local geometric event that can be detected and handled during the sweep. Note that g_i and h_i apply only to f_i and do not affect any other surface f_j, and hence we call them *local dynamic constraints.*

Finally we must slightly modify our kinematic plane sweep. After a pure translational or pure rotational collision with a curve γ_i, we first check to see whether we are in a sticking or sliding region on that curve. If it's a sticking region, we terminate the simulation in equilibrium, otherwise we proceed as above (sec. 3.2). To summarize,

Proposition 3.4 *There are $O(1)$ sticking events per constraint curve γ_i. Each occurs at the intersection of γ_i with a local dynamic constraint (another algebraic curve) in P_Y.*

This completes our proof of the main theorem 2.1.

3.4 Summary

Red-blue merge algorithms allow us to construct a connected component of free space F containing the initial configuration in time $O(\lambda_{s+2}(n)\log^2 n)$. We desire to simulate a simple dynamical

system within F. We compute the simulation trajectory using a plane sweep of F. To do this, we augment F with a vector field (defining an attractor at $u = 0$) and we "annotate" each curve γ_i on the boundary of F with certain "markings" at which the dynamical behavior of a sweep point traversing γ_i can change. The markings break the curve into a finite number of subsegments. The markings are: (a) sticking/sliding transition, (b) vertical tangency, and (c) intersection with the line $u = 0$. Each marking is determined by the intersection of γ_i with a line (such as $u = 0$) or a curve (g_i or h_i), or by vertical tangency. Hence each marking is algebraic and there are $O(1)$ markings per curve. Finally, at an endpoint of γ_i we have an intersection with the next cubic curve γ_j on the boundary of F.

4 Conclusion

We considered the problem of simulating the motion of compliantly connected rigid bodies in frictional contact with obstacles during an assembly motion. While compliant motion has been considered in a computational geometric setting for pure translations [Don2, Eri, FHS], the problem for rotational compliance has proved resistant to solution. We showed that unlike many simulation problems for rotational bodies, we can obtain exact solutions without integration. We improve on our earlier, naïve $O(n^2 \log n)$ algorithm [DP] by the introduction of several techniques. The key ideas we use are: red-blue merge algorithms, a simple dynamical systems model, and local dynamic constraints. These tools permit us to reduce the simulation to a plane sweep of a "dynamically annotated" slice of configuration space. More specifically: First, we precompute the connected component of the simulation. This component of free space has low combinatorial complexity (by Davenport-Schinzel arguments) and can be computed efficiently using a red-blue merge algorithm [GSS]. Next, we reduce the simulation problem to a plane sweep of F. To do this, we first introduce additional local constraints ($O(1)$ per curve bounding F), an attractor at the rest orientation $u = 0$, and a corresponding attractive vector field on F. These constructions allow us to view the plane sweep as a simple dynamical system. This in turn permits us to bound the number of sweep events by $O(\lambda_{s+2}(n))$, which yields our main result. This is one of the first combinatorially efficient, exact solutions to any simulation problem for a rotational mechanical system, or for rotational compliant motion.

There are many problems left open for the future. First, we would like to extend our work to "trees" of compliantly connected bodies. Second, we believe our work can be extended to incorporate uncertainty in the initial conditions and in the control. This result would be of considerable interest, since it would permit simulation of a differential inclusion. Exact simulation does not take into account the uncertainty (eg., the impossibility of a comprehensive description of the dynamics of a system), nor error in actuation. We view the introduction of a more realistic mechanical model (rotationally compliant bodies) as a step in this direction. We feel that the key property that allowed us to reduce the local dynamics to a plane sweep is a kind of "monotonicity" that is inherent in our system. It is our hope that other "monotonic" systems (and even differential inclusions) may be simulated using the concept of "simulation as sweep." See [BD] for work in this direction.

References

[ASS] Agarwal, A., M Sharir and P. Shor *Sharp upper and lower bounds for the length of general davenport-Schintzel sequences,* Tech. Rept. 332, Comp. Sci., Dept., Courant Institute (1987).

[Bri] **Briggs, A.** *An Efficient Algorithm for One-Step Compliant Motion Planning with Uncertainty,* 5^{th} ACM Symposium on Computational Geometry, Saarbrucken, Germany. (1989).
[BD] **Briggs, A. and B. Donald** *Geometric algorithms for simulation of quasi-static mechanical systems under control uncertainty,* Forthcoming (1990).
[Bro] **Brost, R.** *Computing Metric and Topological Properties of Confuguration Space Obstacles,* IEEE Int. Conference on Robotics and Automation, Scottsdale, AZ (1989).
[Can] **Canny, J. F.** *Collision Detection for Moving Polyhedra,* PAMI-8(2) (1986).
[Can2] **John Canny.** *On computability of fine motion plans.* , IEEE ICRA, Scottsdale, AZ, (1989)
[CDRX] **Canny, J., Donald, B., Reif, J., and Xavier, P.** *On The Complexity of Kinodynamic Planning.* 29^{th} Symposium on the Foundations of Computer Science, White Plains, NY pp. 306-316. (1988).
[CR] **Canny, J., and J. Reif,** "New Lower Bound Techniques for Robot Motion Planning Problems". *FOCS* (1987).
[Don] **Donald, B. R.** *A Search Algorithm for Motion Planning with Six Degrees of Freedom,* Artificial Intelligence, 31 (3) (1987).
[Don2] **Donald, B. R.** *The Complexity of Planar Compliant Motion Planning with Uncertainty,* Proc. ACM Symp. Comp. Geom., Urbana (Also *Algorithmica,* 5 (3), 1990 pp. 353-382.) (1988).
[Don3] **Donald, B. R.** *Planning Multi-Step Error Detection and Recovery Strategies,* Int. Jour. Robotics Research, 9 (1), pp. 3-60. (1990).
[DP] **Donald, B. and D. Pai.** *On The Motion of Compliantly-Connected Rigid Bodies in Contact: A System for Analyzing Designs for Assembly,* Proc. IEEE ICRA, Cincinatti (1990).
[Erd] **Erdmann, M.** *Using Backprojections for Fine Motion Planning with Uncertainty,* IJRR Vol. 5 no. 1 (1986).
[FHS] **Friedman, J., J. Hershberger and J. Snoeyink** *Compliant Motion in a Simple Polygon,* 5^{th} ACM Symposium on Computational Geometry, Saarbrucken, Germany. (1989).
[GSS] **Guibas, L., Sharir, M., and Sifrony, S.** *On the general motion planning problem with two degrees of freedom,* Proc. ACM Symp. Comp. Geom., Urbana, Il (1988).
[HS] **H: t, S., and Sharir, M.** *Nonlinearity of Davenport Schintzel sequences and of generalized path compression,* Combinatorica, (2) pp. 209-233 (1987).
[KS] **Kedem, K., and Sharir, S.** *Efficient algorithms for planning translational collision-free motion of a convex polygonal object in 2-dimensional space amidst polygonal obstacles,* Proc. ACM. Symp. Comp. Geom., pp 75-80 (1985).
[LoP] **Lozano–Pérez, T.** *Spatial Planning: A Configuration Space Approuch,* IEEE Trans. on Computers (C–32):108–120 (1983a).
[LMT] **Lozano-Pérez, T., Mason, M. T., and Taylor, R. H.** *Automatic Synthesis of Fine-Motion Strategies for Robots,* Int. J. of Robotics Research, Vol 3, no. 1 (1984).
[Ma] **Mason, M. T.** 1986. Mechanics and Planning of Manipulator Pushing Operations. *International Journal of Robotics Research* 5(3).
[PD] **Pai, D. and B. Donald** *On The Motion of Compliantly-Connected Rigid Bodies in Contact, Part I: The Motion Prediction Problem,* Cornell Computer Science Technical Report, 89-1047 (1989).
[Whi] **Whitney, D.,** "Quasi-Static Assembly of Compliantly Supported Rigid Parts", *Journal of Dynamic Systems, Measurement, and Control,* Vol. 104, March 1982.
[Yap] **Yap, C.,** "Algorithmic Motion Planning", in *Advances in Robotics: Volume 1,* edited by J. Schwartz and C. Yap, Lawrence Erlbaum Associates, 1986.

Acknowledgments

We are very grateful to John Canny for suggestions and advice, which were invaluable to us in obtaining our results. Also many thanks to Randy Brost for kindly providing us with illustrations of configuration space.

APPENDIX: FIGURES

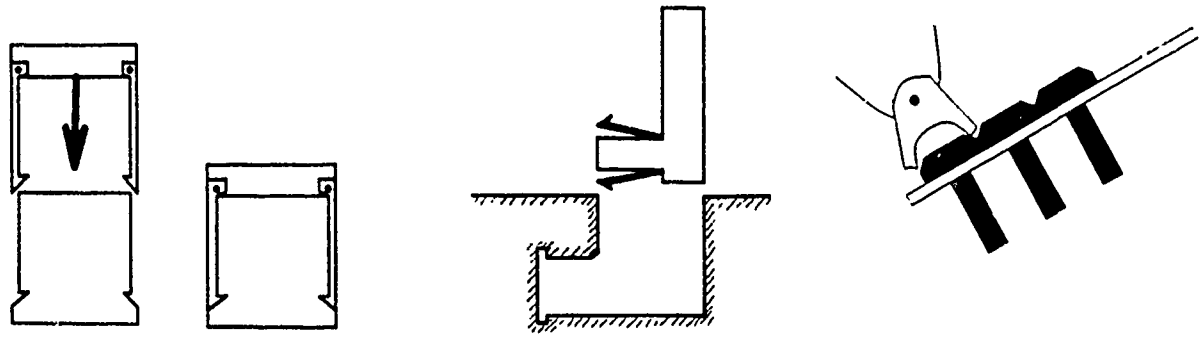

Figure 1: Examples of compliantly connected rigid bodies.

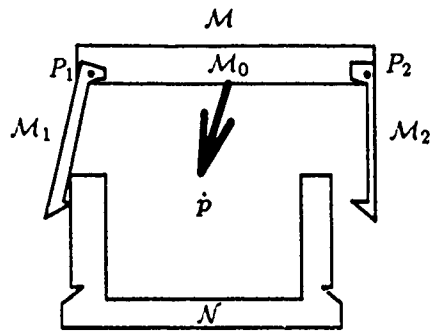

Figure 2: Linked body $\mathcal{M}$ moving among $\mathcal{N}$.

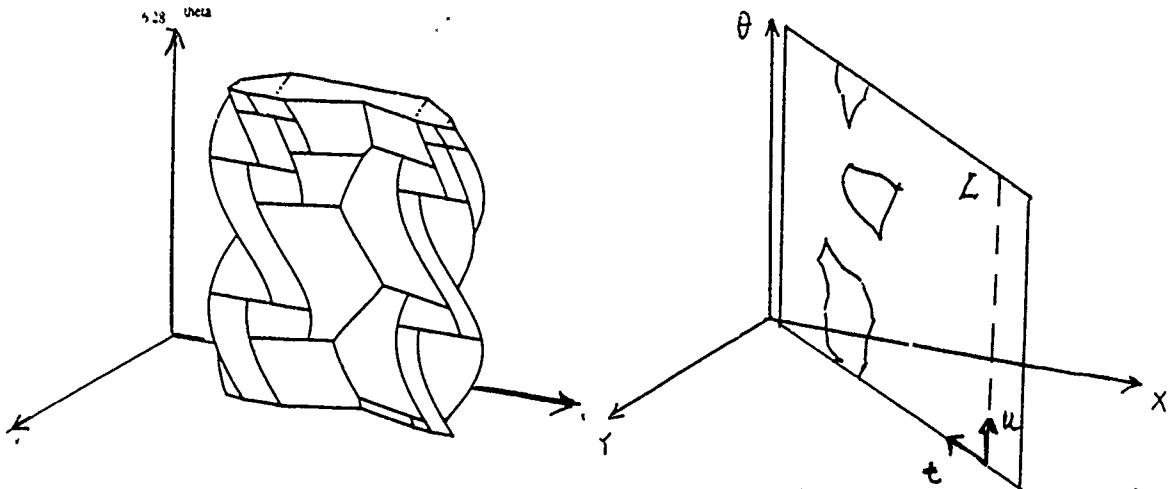

Figure 3: Configuration space constraints for a moving pawl. (Reprinted courtesy of Randy Brost [Bro]). The sweep plane $L(t)$ intersects the cspace obstacles in a planar arrangement of cubic curves.

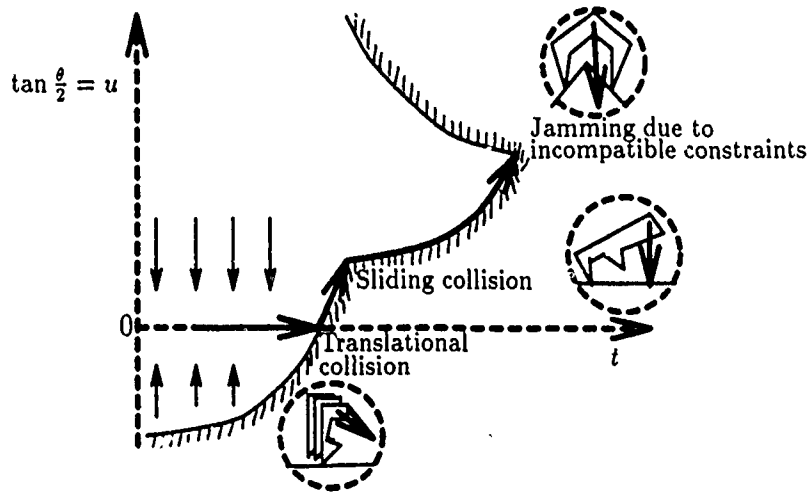

Figure 4: Sweep events: translational collision, sliding collision, and jamming due to incompatible kinematics.

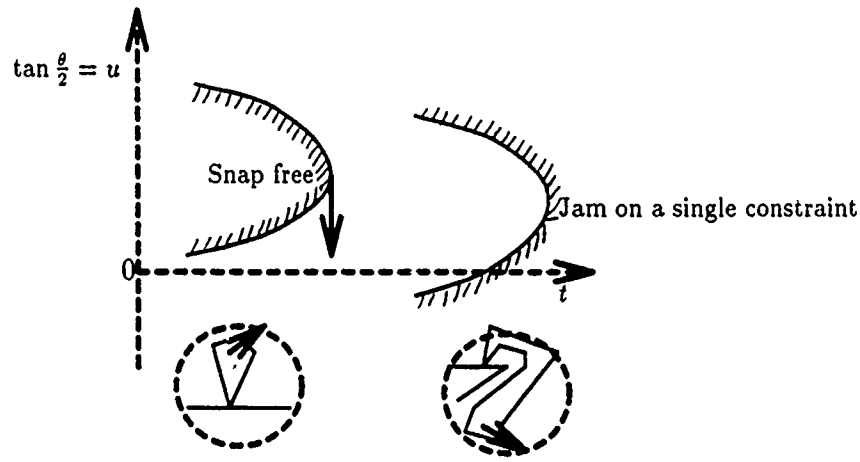

Figure 5: Sweep events: Snapping free and jamming on a single constraints.

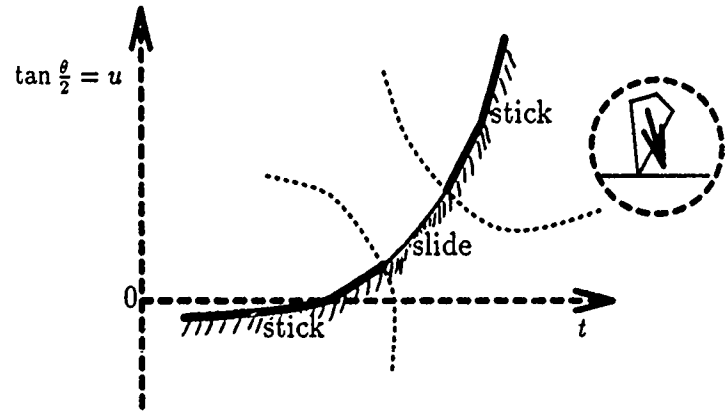

Figure 6: Sticking events and qualitative dynamical regions.